IFW 2624

PATENT APPLICATION

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of	Docket No: Q57419
Shunsaku MIYAZAWA, et al.	Allowed: March 10, 2005
Appln. No.: 09/531,214	Group Art Unit: 2624
Confirmation No.: 5963	Examiner: Douglas Q. TRAN
Filed: March 20, 2000	Issued: Not yet issued
U.S. Patent No.: Not yet issued	

For: DATA COMMUNICATION APPARATUS FOR RESUMING DATA TRANSFER AFTER INTERRUPTION

SUBMISSION OF ART

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

For the possible benefit of anyone subsequently evaluating the scope and/or validity of the above-identified patent, it is requested that the document that is listed below (copy enclosed) be placed in the U.S. Patent and Trademark Office's file wrapper of the above-identified U.S. patent:

"Information Technology-Serial Bus Protocol 2, Working Draft, Revision 2g" dated September 15, 1997, pgs. 13, 16, 17, 67, 82, 83.

The above-listed document (D3) was recently cited in a communication from a Foreign Patent Office dated April 28, 2005. Documents D1 and D2, also listed in the communication, were previously cited in an Information Disclosure Statement.

The undersigned has not reviewed the teachings of the above-listed document in detail and thus makes no representations concerning the relevancy or materiality of the above-listed document.

This is not an Information Disclosure Statement and no response from the U.S. Patent and Trademark Office is believed to be necessary, nor are any fees believed to be due.

Respectfully submitted,

Allison M. Tulino
Registration No. 48,294

SUGHRUE MION, PLLC
Telephone: (202) 293-7060
Facsimile: (202) 293-7860

WASHINGTON OFFICE
23373
CUSTOMER NUMBER

Date: July 7, 2005

EPA/EPO/OEB
D-80298 München
☎ +49 89 2399-0
TX 523 656 epmu d
FAX +49 89 2399-4465

Europäisches Patentamt	European Patent Office	Office européen des brevets
Generaldirektion 2	Directorate General 2	Direction Générale 2

Telephone numbers:

Primary Examiner (substantive examination)	+49 89 2399-8245
Formalities Officer / Assistant (Formalities and other matters)	+49 89 2399-7574

Sturt, Clifford Mark
Miller Sturt Kenyon
9 John Street
London WC1N 2ES
ROYAUME-UNI

Application No.	Ref.	Date
00 302 190.4 - 2211	EPP13342A	28.04.2005

Applicant
SEIKO EPSON CORPORATION

Communication pursuant to Article 96(2) EPC

The examination of the above-identified application has revealed that it does not meet the requirements of the European Patent Convention for the reasons enclosed herewith. If the deficiencies indicated are not rectified the application may be refused pursuant to Article 97(1) EPC.

You are invited to file your observations and insofar as the deficiencies are such as to be rectifiable, to correct the indicated deficiencies within a period

of 4 months

from the notification of this communication, this period being computed in accordance with Rules 78(2) and 83(2) and (4) EPC.

One set of amendments to the description, claims and drawings is to be filed within the said period on separate sheets (Rule 36(1) EPC).

Failure to comply with this invitation in due time will result in the application being deemed to be withdrawn (Article 96(3) EPC).

Europäisches Patentamt · European Patent Office · Office européen des brevets

Fournier, N
Primary Examiner
for the Examining Division

Enclosure(s): 5 page/s reasons (Form 2906)
Document D3: Serial Bus Protocol 2

1 Reference is made to the following documents

D1: EP-A-0 820 185 (MATSUSHITA ELECTRIC IND CO LTD) 21 January 1998 (1998-01-21)

D2: US-A-5 038 299 (MAEDA YUJI) 6 August 1991 (1991-08-06)

1.1 The following document (**D3**) is cited by the examiner (see the Guidelines, C-VI, 8.7). A copy of the document is annexed to the communication and the numbering will be adhered to in the rest of the procedure:

D3: "Information technology - Serial Bus Protocol 2, Working Draft, Revision 2g", dated 15 September 1997, pages 13, 16, 17, 67, 82, 83. Available online at http://www.t10.org/ftp/t10/drafts/sbp2/sbp2r02g.pdf

Clarity

2 The application does not meet the requirements of Article 84 EPC, because claims **1-3** are not clear.

2.1 In claim **1**, on line 13, it is unclear to what the term "at a time" refers to: the "presenting" or the "status". In light of the description (page 5, lines 23-25, and page 7, lines 9-20), line 13 is interpreted as "presenting an execution status of said job with respect to the time when the execution".

2.2 In claim **1**, on lines 15-16, it is unclear to what the term "at a time" refers to: the word "refers" or the word "status". In light of the description (page 5, lines 23-25, and page 7, lines 9-20), line 16 is interpreted as "status of said job with respect to the time when the execution of the job was".

2.3 In claim **2**, on line 9, the phrasing "said job execution means follows ORB" does not make sense, as a job execution means cannot literally "follow" an ORB. In light of the description (page 17, lines 11-20) this is interpreted as "said job execution means uses the information given in an ORB".

2.4 In claim **2**, line 29, the word "been" appears to be missing between the words "already" and "obtained".

2.5 In claim **3**, lines 9-14, the phrasing is unclear, leaving the reader in doubt as to what is meant. These lines have been interpreted as follows:
"and at least one of: the number of sent Read Block Requests generated to execute said ORB, the number of Read Block Responses received from the partner apparatus as responses to said Read Block Requests, and the total byte number of read data contained in said Read Block Responses; and".

3 Notwithstanding the objections raised above, and for the sake of efficiency of the procedure, a preliminary opinion regarding novelty and inventive step is nevertheless given for all claims, where the claims have been interpreted as mentioned above.

Novelty and Inventive Step

4 The subject-matter of claim 1 does not meet the requirements of Article 54 (1) EPC for the following reasons:

4.1 Document **D1** discloses the subject-matter of claim 1 as follows:

"A data communication apparatus for performing data communication with a partner apparatus through a communication line, said data communication apparatus comprising (**D1**, column 4, line 42-column 5, line 11: *the digital multi-function printer corresponds to the data communication apparatus, and the computer corresponds to the partner apparatus; they are connected via a network and communicate with each other, as the printer receives data from the computer*):

- job execution means for receiving data from the partner apparatus, for executing a job; and
- job management means for managing an execution status of said job (**D1**, column 5, lines 1-11: *the print queue controller with reference numeral 7 corresponds to both, the job execution means and the job management means*);

wherein:

- when a job whose execution was interrupted by a given event is to be resumed, said job management means instructs said job execution means to resume the job, while presenting an execution status at a time when the execution of the job was interrupted; and
- said job execution means refers to the execution status at the time when the execution of the job was interrupted, for receiving only data required for executing a non-processed part of the job, said execution status being presented by said job management means, relating to the job about which the instruction of resuming has been given (**D1**, column 7, lines 8-35: *lower priority jobs are interrupted by higher priority jobs; the print queue controller maintains the print control data concerning which pages of the lower priority job have already been printed, corresponding to the job's execution status; once the higher priority job is executed, the lower priority job is resumed starting from the first page that has not yet been printed*)."

4.2 Similarly, document **D3** discloses the subject-matter of claim **1** as follows:
"A data communication apparatus for performing data communication with a partner apparatus through a communication line, said data communication apparatus comprising (**D3**, page 13, lines 1-5, 20-25: *the SBP-2 protocol is designed to permit communication between devices via the IEEE 1394 serial bus interface*):
- job execution means for receiving data from the partner apparatus, for executing a job; and
- job management means for managing an execution status of said job (**D3**, pages 16-17, paragraph '4.5 Target agents', and pages 82-83, paragraph '10.5 Task management event matrix': *the stream control target fetch agent corresponds to the job execution means; the device performs task management to control the state of the target fetch agents, implicitly using task management means for this*);
wherein:
- when a job whose execution was interrupted by a given event is to be resumed, said job management means instructs said job execution means to resume the job, while presenting an execution status at a time when the execution of the job was interrupted; and
- said job execution means refers to the execution status at the time when the execution of the job was interrupted, for receiving only data required for executing a non-processed part of the job, said execution status being presented by said job management means, relating to the job about which the instruction of resuming has been given (**D3**, page 67, paragraph '8.3 Reconnection', pages 82-83, paragraph '10.5 Task management event matrix': *the state information of tasks that have been interrupted by a bus reset are maintained for some time, in particular for stream tasks the current stream position is saved, so that when the initiator reconnects the fetching operations can resume from the same point as before the bus reset, i.e. only not-yet-processed data is sent after the bus reset; the task management means take care of implementing this behaviour*)."

COMMENTS

5 The subject-matter of claim **2** might constitute the basis for a new independent claim meeting the requirements or Article 52 (1), for the following reasons.

5.1 The additional subject-matter of claim 2, new over **D1**, **D2**, and **D3** is directed towards recognising a request for a second data transfer over an IEEE 1394 connection as refering to the same data as a previous, first data transfer over the

same IEEE 1394 connection, when the data connection is not a data stream. This has the technical effect of allowing resuming an interrupted data transfer over an IEEE 1394 connection, despite system reconfiguration, when the transfer required an explicit request for data at a certain memory location (i.e., in the phrasing of D3, data requested using a 'normal command block').
This provides the advantage, that in case where a first transfer of data from an explicitly specified area of memory over an IEEE 1394 connection is interrupted due to system reconfiguration before all required data is received, the receiver is able to recognise a request for a second transfer of data from an explicitly specified area of memory as refering to the same area of memory as the first one, and to decide from where to restart data transfer, avoiding the repeated transfer of the already received data.
The objective problem solved by claim **2** may therefore be formulated as how to provide efficient recovery after system reconfiguration has interrupted a transfer of data from a specified area of memory over an IEEE 1394 connection.

5.2 The prior art document **D1**, discloses the interruption of a data transfer, maintaining a record of where the data transfer has been interrupted and later resuming that data transfer by transmitting the data that had not yet been sent. Moreover, the "Serial Bus Protocol 2" is a well known protocoll (see for example the document **D3**), designed for connection of devices over IEEE 1394.

5.3 The prior art document **D3**, discloses the interruption of a streamed data transfer, maintaining a record of where the streamed data transfer has been interrupted and later resuming that streamed data transfer by transmitting the data that had not yet been sent. However, when it comes to non-streamed data, document **D3** suggests that interrupted transfers should be aborted entirely (**D3**, page 83, line16-17), thereby leading the skilled person away from the solution put forward in claim **2**.

5.4 The cited prior art did not reveal anything that would point the person skilled in the art of data communication to modify **D1** or **D3**, so as to arrive at the solution disclosed in claim **2**.

5.5 Therefore, the subject matter of claim **2** would constitute a basis for subject-matter meeting the requirements of Article 52 (1) EPC.

6 It is suggested that a new independent claim should be filed based on the combination of all features contained in claims **1** and **2**, taking into account any objections put forward under "Clarity" and "Form and Content".

7 In order to facilitate the examination of the conformity of the amended application

with the requirements of Article 123(2) EPC, the applicant is requested to clearly identify the amendments carried out, no matter whether they concern amendments by addition, replacement or deletion, and to indicate the passages of the application as filed on which these amendments are based. These indications should be submitted in handwritten form on a copy of the relevant parts of the application as filed.

Form and Content

8 Document **D3** is considered the closest prior art. Any independent claim should be in the two-part form with respect to document **D3**, to meet the requirements of Rule 29(1) EPC.

9 The features of the claims should be provided with reference signs placed in parentheses (Rule 29(7) EPC).

9.1 Note that in figure 6, the lines linking reference numbers 2151 and 2152 to the execution status monitoring part and to the job execution part, respectively, appear not to be drawn correctly. These two parts appear to correspond to the job management means and job execution means, respectively, of claim 1. As figure 6 is the main reference figure, and any reference numerals introduced into the independent claims would relate to this figure, this inaccuracy in the figure would render the claims unclear.
Therefore, figure 6 should be corrected, otherwise it would lead to a clarity objection under Article 84 EPC, once the requirements of Rule 29(7) EPC are fulfilled.

10 In order to put the invention into proper perspective, documents **D1-D3** should be identified in the description, and the relevant background art disclosed therein should be discussed (Rule 27(1)(b) EPC and Guidelines C-II, 4.3).. Care should be taken not to introduce new subject-matter in the process (Article 123(2) EPC).

Working Draft

T10
Project 1155D

Revision 2g
September 15, 1997

Information technology —
Serial Bus Protocol 2 (SBP-2)

This is a draft proposed American National Standard under development by T10, a Technical Committee of the National Committee for Information Technology Standardization (NCITS). As such, this is not a completed standard and has not been approved. The Technical Committee may modify this document as a result of comments received during public review and its approval as a standard.

T10 Technical Editor:

Peter Johansson
Congruent Software, Inc.
3998 Whittle Avenue
Oakland, CA 94602
USA

(510) 531-5472
(510) 531-2942 FAX

pjohansson@aol.com

Reference numbers
ISO/IEC xxxxx:199x
ANSI NCITS.xxx-199x

Printed September 15, 1997

Points of contact

T10 Chair:

John B. Lohmeyer
Symbios Logic, Inc.
4420 Arrows West Drive
Colorado Springs, CO 80907
USA

(719) 533-7560
(719) 533-7036 FAX
john.lohmeyer@symbios.com

T10 Vice-chair:

Lawrence J. Lamers
Adaptec, Inc.
691 South Milpitas Boulevard
Milpitas, CA 95035

(408) 957-7817
(408) 957-7193 FAX
ljlamers@aol.com

NCITS Secretariat:

NCITS Secretariat
1250 I Street NW, Suite 200
Washington, DC 2000
USA

(202) 737-8888
(202) 638-4922 FAX

T10 Bulletin board:

(719) 533-7950

T10 FTP:

ftp.symbios.com/pub/standards/io/x3t10

T10 Home page:

http://www.symbios.com/x3t10

T10 Reflector:

scsi@symbios.com
majordomo@symbios.com (to subscribe)

IEEE 1394 Reflector:

p1394@sun.com
bob.snively@sun.com (to subscribe)

Document distribution:

Global Engineering
15 Inverness Way East
Englewood, CO 80112-5704
USA

(800) 854-7179
(303) 792-2181
(303) 792-2192 FAX

ANSI®
NCITS.xxx-199x

American National Standard
for Information Systems –

Serial Bus Protocol 2 (SBP-2)

Secretariat

Information Technology Industry Council

Not yet approved

American National Standards Institute, Inc.

Abstract

This standard specifies a protocol for the transport of commands, data and status between devices connected by Serial Bus, a memory-mapped split-transaction bus defined by IEEE Std 1394-1995. In order to take advantage of unique capabilities of Serial Bus for the transport of isochronous data, this standard provides methods to manage isochronous connections and to control the flow of isochronous data between devices.

American National Standard

Approval of an American National Standard requires verification by ANSI that the requirements for due process, consensus and other criteria for approval have been met by the standards developer.

Consensus is established when, in the judgment of the ANSI Board of Standards Review, substantial agreement has been reached by directly and materially affected interests. Substantial agreement means much more than a simple majority, but not necessarily unanimity. Consensus requires that all views and objections be considered and that effort be made towards their resolution.

The use of American National Standards is completely voluntary; their existence does not in any respect preclude anyone, whether he has approved the standards or not, from manufacturing, marketing, purchasing, or using products, processes, or procedures not conforming to the standards.

The American National Standards Institute does not develop standards and will in no circumstances give interpretation on any American National Standard. Moreover, no person shall have the right or authority to issue an interpretation of an American National Standard in the name of the American National Standards Institute. Requests for interpretations should be addressed to the secretariat or sponsor whose name appears on the title page of this standard.

CAUTION NOTICE: This American National Standard may be revised or withdrawn at any time. The procedures of the American National Standards Institute require that action be taken periodically to reaffirm, revise, or withdraw this standard. Purchasers of American National Standards may receive current information on all standards by calling or writing the American National Standards Institute.

CAUTION NOTICE: The developers of this standard have requested that holder's of patents that may be required for the implementation of this standard, disclose such patents to the publisher. However, neither the developers nor the publisher has undertaken a patent search in order to identify which, if any, patents may apply to this standard.

Published by

American National Standards Institute
1430 Broadway, New York, NY 10018

Printed in the United States of America

4 Model (informative)

Serial Bus Protocol 2 (SBP-2) is a transport protocol defined for IEEE Std 1394-1995, Standard for a High Performance Serial Bus. It defines facilities for requests (commands) originated by Serial Bus devices (initiators) to be communicated to other Serial Bus devices (targets) as well as the facilities required for the transfer of data or status associated with the commands. An SBP-2 device may assume both roles, initiator or target, either simultaneously or in succession. Commands and status may be transferred between the initiator and the target; data moves between the target and another device, which may be neither the initiator nor an SBP-2 device.

This clause is informative and describes components of the SBP-2 model. It is intended to enhance the usefulness of the other, normative parts of this standard. In addition to the information in this clause, users of this standard should also be familiar with the CSR architecture and Serial Bus standards.

4.1 Unit architecture

In CSR architecture and Serial Bus terminology, devices implemented to this standard (targets) are units. A Serial Bus node that implements a target has a unit directory in configuration ROM that identifies the presence and capabilities of the target.

The unit directory in configuration ROM permits initiators to detect the presence of targets during Serial Bus configuration, whether part of system initialization or subsequent to a Serial Bus reset. The node's 64-bit identifier, EUI-64, permits detected targets to be uniquely recognized despite changes in physical addresses that may occur as the result of Serial Bus resets.

4.2 Logical units

A logical unit is part of the unit architecture and is an instance of a device model, *e.g.*, mass storage, CD-ROM or printer. A logical unit consists of one or more device server(s) responsible to execute control or data transfer commands, zero or more stream controllers, one or more task sets that hold commands available for execution by the device server(s) or stream controller(s) and a logical unit number that is unique within the domain of the target.

Targets implement at least one logical unit, addressable as logical unit number (LUN) zero. Additional logical units may be implemented, which may be addressable by their logical unit numbers. The logical units may implement different device models; for example, a single unit architecture might contain both a CD-ROM logical unit and an associated medium-changer logical unit. The logical unit(s) are visible to the initiator, either as described by configuration ROM or as discoverable by command set-dependent requests directed to the target.

4.3 Requests and responses

Target actions, such as a disk read that transfers data from device medium to system memory, are specified by means of requests created by the initiator and signaled to the target. The request is contained within a data structure called an operation request block (ORB). The eventual completion status of a request is indicated by means of a status block stored by the target at an address provided by the initiator.

This standard defines several different formats for request blocks, whose principal uses are:

- to obtain access to target resources (login requests);
- to transport command blocks (normal and stream command block requests);
- to manage task sets or to release target resources (management requests); or
- to control the flow of isochronous data (stream control requests).

The fields in the ORB that directly addressed a data buffer in the first example now point to a page table. Note that the ORB field that contains the data length when direct addressing is employed instead contains the number of elements in the page table—in this case, four. Each of the four page table elements points to the start of a segment of the data buffer. Each page table element also contains the length of the segment. The first segment ends on a page boundary, all other segments start on page boundaries (and the middle segments also end on page boundaries) while the last segment may end on any boundary. In this example, the segment lengths are 0564_{16}, 1000_{16}, 1000_{16} and $03FC_{16}$, respectively.

When a page table is used, both the page table and the data buffer it describes reside in the same node. The node ID of the page table, $FFC0_{16}$, is not repeated in the page table elements. The space that would have otherwise been occupied by the node ID instead is used to contain the length of each segment.

Another variant of page table format is permitted, called an unrestricted page table (or a scatter/gather list). In an unrestricted page table, data buffer segments may start on any boundary and may have arbitrary lengths: there is no underlying page size.

4.5 Target agents

A target agent is a facility that receives signals from the initiator that indicate the availability of requests. There are two types of target agent, one that can execute a single request at a time and the other that can manage queues (linked lists) of requests, as illustrated by Figure 6. In the first case, the initiator signals the request to the agent by means of a Serial Bus block write request with the address of the request. In the other case, the initiator appends new requests to an active list, rings a doorbell which causes the target agent to fetch the requests from system memory as target resources permit their execution.

Target agents that manage linked lists of requests utilize context maintained at both the initiator and target to fetch requests from memory. Once fetched, the request is locally available to the target for execution. The context consists of three elements:

- a linked list of ORB's at the initiator;
- a current ORB address at the target; and
- a doorbell at the target.

This standard defines procedures for both the initiator and the target that permits the addition of new requests to a linked list of ORB's while the target is actively fetching or executing previously enqueued requests. The procedures avoid the possibility of race conditions between the producer (initiator) and consumer (target) of the ORB's.

There are three defined target agents:

- management;
- command block; and
- stream control.

Management agents accept a variety of requests: login, create stream, task management and logout. Before making other requests, an initiator first completes a login *via* the management agent. Once this is done, the management agent will accept create stream requests and task management requests. The latter are directed to either a normal (asynchronous) task set or to a task set associated with an isochronous stream. Ultimately, management agents accept logout requests; these indicate the initiator's intent to release target resources previously acquired by a login or create stream request. Management agents service a single request at a time and do not support linked lists.

A successful login or create stream request returns the address of a command block agent. Command block agents service either normal or stream command block requests that are organized into linked lists. Each linked list is managed by a separate command block agent.

A create stream request may return the address of a stream control agent. This agent meters the flow of isochronous data to or from Serial Bus. Unless the target provides other facilities to meter this flow (*e.g.*, plug control registers as specified by IEEE P1394a), each stream requires both a stream command block agent and a stream control agent to coordinate operations. The time-critical nature of isochronous operations requires that stream control agents support linked lists of requests, just as command block agents.

4.6 Ordered and unordered execution

Targets may implement either an ordered or unordered model of task execution. The ordered model is usually appropriate for devices where the context of a command affects its execution, *i.e.*, the outcome of one command affects the subsequent command. A common example of a device with such command dependencies is a tape drive. The unordered model is usually appropriate for devices, such as mass storage, where no positional or other context information is inherited from one command to the next.

The ordered model specifies both that tasks shall be executed in order and that completion status shall be returned in the same order. A consequence of ordering that completion status for one task implicitly indicates successful completion status for all tasks that preceded it in the ordered list.

The unordered model permits the target to reorder active tasks without restriction. The actual execution sequence of tasks from any task set may bear no relationship to the order in which they were fetched. Unrestricted reordering places the responsibility for the assurance of data integrity on the initiator. If the integrity of data on the device medium could be compromised by unrestricted reordering involving a set of active tasks, $\{T_0, T_1, T_2, \ldots T_N\}$ and a new task T´, the initiator shall wait until $\{T_0, T_1, T_2, \ldots T_N\}$ have completed before appending T´ to an active request list.

NOTE – In multitasking operating system environments, independent execution threads may generate tasks that have ordering constraints within each thread but not with respect to other threads. If this is the case, an initiator may manage the constraints of each thread yet still keep the target substantially busy. This avoids the undesirable latencies that occur if the target is allowed to become idle before new ORB's are signaled.

4.7 Streams

Streams are objects that are based upon the isochronous capabilities of Serial Bus. A stream consists of all of the target functions and resources that are necessary to transfer isochronous data from one or more Serial Bus channels to the device's medium (the target is a listener) or to transfer isochronous data from the device's medium to one or more Serial Bus channels (the target is a talker). The direction, listener or talker, of any stream is independent of any other stream. Within each stream all of the channels flow in the same direction.

Streams require Serial Bus resources as well as target resources. These include the aggregate bandwidth necessary for the stream, the channel numbers utilized by the stream and the isochronous connections that characterize the stream. An application allocates all necessary resources before activating a target isochronous stream.

A stream of isochronous data appears on Serial Bus as packet(s) during each isochronous cycle. This in turn is represented by an ordered byte stream of data on the device medium. The presentation of this data is controlled by ORB's that request data transfer to or from the medium and, optionally, other ORB's that control its flow on Serial Bus. The second type of ORB, the stream control ORB is not required if the target has other facilities to control the flow of isochronous data from or to Serial Bus.

The target shall perform the following to validate a create stream request:

a) The target shall validate the *login_ID* supplied in the create stream ORB by comparing the *destination_ID* in the read request(s) used to fetch the ORB with the *source_ID* retained when *login_ID* was assigned to the initiator. If the node ID's do not match, the *login_ID* is invalid.

b) If the *login_ID* is valid, the target shall determine if a free *login_descriptor* is available. If a *login_descriptor* is free, the initiator's *source_ID* is stored in *login_owner_ID*, the initiator's EUI-64 is stored in *login_owner_EUI_64*, the *lun* from the *login_descriptor* is copied to the *login_descriptor* for the create stream request and the addresses of the fetch agent(s) are also stored in the *login_descriptor*. Lastly the target assigns a unique *login_ID* to this login and stores it in the *login_descriptor*.

In addition to the addresses of the stream command block and stream control fetch agents, the target shall also specify in the *login_response* data the minimum transfer length that the initiator should specify in the *stream_length* field of any stream command block request signaled to the target.

8.3 Reconnection

Upon a Serial Bus reset, the target shall abort all task sets for all command block agents created as the result of login request(s). Task sets associated with isochronous streams shall not be aborted. Both the stream command block and stream control requests shall continue to be executed by the target but the return of status shall be deferred until a successful reconnection.

For one second subsequent to a bus reset the target shall retain sufficient information to permit an initiator to reconnect its login ID and associated stream ID's. After one second the target shall perform an implicit logout for all login ID's and stream ID's that have not been successfully reconnected to their original initiator(s).

NOTE – The basis of the one second time-out is to permit initiators to reallocate isochronous channels and bandwidth and to reestablish isochronous connections. The time-out commences when the target observes the first subaction gap subsequent to a bus reset; if a bus reset occurs before the time-out expires, the timer is zeroed then restarted upon detection of a subaction gap.

The target shall perform the following to validate a reconnect request:

a) The target shall read the initiator's unique ID, EUI-64, from the bus information block by means of two quadlet read transactions. The *source_ID* from the write transaction used to signal the reconnect ORB to the target's MANAGEMENT_AGENT register shall be used as the *destination_ID* in the quadlet read transactions;

b) The target shall determine whether or not the *login_ID* is valid by comparing the just obtained EUI-64 against the *login_owner_EUI_64* for the *login_descriptor* identified by *login_ID*;

c) If the *login_ID* is valid, the target shall store the initiator's *source_ID* in *login_owner_ID* for the referenced login_descriptor and for all stream descriptors associated with the same initiator; and

d) Fetch agents for stream command block and stream control requests for the reconnected initiator may resume; status for completed ORB's that had not been stored in the initiator's *status_FIFO* (because the initiator's *source_ID* had been invalidated by the bus reset) may also be stored.

No *login_response* data is stored for a reconnect request; the completion status is indicated by the status block stored at the *status_FIFO* address.

8.4 Logout

When an initiator no longer requires access to a target's resources, it shall signal a logout request to the management agent. The login or stream resources to be released shall be identified by *login_ID* in the

c) The target shall create a unit attention condition for all logged-in initiators other than the initiator, identified by *login_ID*, that signaled the target reset request; and

d) When all of the above events have completed, the target shall store completion status for the target reset request in the status buffer provided. The completion status shall indicate FUNCTION COMPLETE.

The initiator shall not reuse the system memory occupied by any of the affected ORB's, data buffers or page tables of the tasks until completion status is returned for the target reset request.

10.4.6 Terminate task

Terminate task is a task management function that permits an initiator to request early completion of a specified task. Targets that implement the basic task management model shall not support terminate task and shall reject all terminate task requests with a completion status of FUNCTION REJECTED.

To request task termination, the initiator shall construct a management ORB in system memory for the terminate task function. The initiator shall set the appropriate values in the *rq_fmt*, *login_ID* and *ORB_offset* fields of the ORB, as described in 5.1.4.6. The *function* field shall be set to TERMINATE TASK; *ORB_offset* shall contain the Serial Bus address of the ORB for the task to be terminated. Once the terminate task ORB has been initialized, the initiator shall signal the ORB to the management agent.

Upon receipt of a terminate task request, the target shall store a completion status of FUNCTION COMPLETE or FUNCTION REJECTED for the terminate task request in the status buffer provided.

If the terminate task function is accepted by the target, the completion status of FUNCTION COMPLETE does not necessarily indicate that the specified task has completed. The ultimate completion of the specified task shall be signaled when the target stores completion status for the task.

If an error condition is detected for the specified task, the terminate task request shall be ignored and the target shall perform the actions previously described in 10.3.

If the specified task completes prior to the receipt of the terminate task request, the target shall wait until completion status is successfully stored for the specified task before completion status shall be stored for the terminate task request.

Otherwise, the target shall complete the specified task as follows:

a) The target shall not issue data transfer requests for the task;

b) The target shall wait for responses to pending data transfer requests;

c) The target shall store completion status of REQUEST COMPLETE and appropriate command set-dependent status that indicates command termination.

When a terminated task creates an error condition, the target shall clear the task set and take the actions described in 10.3.

The initiator shall not reuse the system memory occupied by the ORB, data buffer or page table of the task to be terminated until completion status is returned for that ORB.

10.5 Task management event matrix

Common events that affect the state of target fetch agents and their associated task set(s) are summarized below. Refer to the governing clauses in sections 8 and 9 as well as this section for detailed information.

	AGENT_STATE.*st*		Task set(s)	
Event	**Normal**	**Stream**	**Normal**	**Stream**
Power reset	RESET		Clear all task sets	
Command reset (write to RESET_START)	RESET		Clear all task sets	
Bus reset (immediate)	RESET	—	Clear all task sets	—
Bus reset (after one second)	—		Logout any initiator that has failed to successfully reconnect	
Login	—		—	
Create stream	—		—	
Reconnect	—		—	
Logout	RESET		Abort initiator's task set	
Faulted command (CHECK CONDITION)	DEAD		Abort faulted initiator's task set	
ABORT TASK	—		—	
ABORT TASK SET	DEAD		Abort initiator's task set	
CLEAR TASK SET	DEAD		Clear all task sets	
LOGICAL UNIT RESET	DEAD		Abort all the logical unit's task sets	
TARGET RESET	DEAD		Clear all task sets	
TERMINATE TASK	—		—	

With respect to events supported by the target's management agent, *e.g.*, logout, there is an assumption of successful completion. In the case of a function rejected response or other indication of failure, the preceding table does not apply.

Bus resets affect target fetch agents and task sets according to the kind of request, login or create stream, by which the initiator first acquired access privileges. A login request allocates normal command block resources while a create stream request allocates stream command block and stream control resources.

Immediately upon detection of a bus reset, all normal command block fetch agents transition to the reset state and their associated task sets are cleared. Stream command block and stream control fetch agents do not fetch any additional ORB's subsequent to a bus reset but otherwise preserve their state. The task sets associated with these stream agents continue execution, but status for completed commands is held by the target and not stored to the initiator's *status_FIFO*.

For one second subsequent to a bus reset, targets save state information for initiators that were logged-in at the time of the bus reset. The one-second timer commences when the target observes the first subaction gap subsequent to a bus reset; if a bus reset occurs before the timer expires, the timer is reset. If an initiator successfully completes a reconnect request during this period, the actions described in 8.3 occur. For normal command block requests, the task set is empty and the initiator may signal new ORB's to the target. For both stream command block and stream control agents, fetching operations resume from the same point as before the bus reset. Any completion status held by the target during this one second period may also be stored to the initiator's *status_FIFO* after the successful reconnection.

One second after a bus reset, the target shall automatically perform a logout operation for all login ID's and stream ID's that have not been reconnected with their initiator. This returns all the affected fetch agents to the reset state and aborts any associated stream task sets.